SELECTION AND ENHANCEMENT OF GABOR FILTERS FOR AUTOMATIC SPEECH RECOGNITION[†]

György Kovács[1], László Tóth[1], Dirk Van Compernolle[2]
[1] *MTA-SzTE Research Group on Artificial Intelligence, Hungarian Academy of Sciences, Szeged, Hungary*
[2] *Department of Electrical Engineering (ESAT), KU Leuven, Heverlee, Belgium*

Abstract. Motivated by neurophysiological studies, the use of Gabor filters as acoustic feature extractors for speech recognition purposes has received increasing attention in the new millenium. As the optimal parametrization of these filters is not obvious, many researchers employ different feature selection methods to find the best filter set. In this study, however, we argue that these kinds of feature selection methods cannot fulfill this task, as we demonstrate this with results obtained from experiments. We show that one can easily construct a better filter set manually, using simple heuristic rules. Then, as an alternative to the usual filter selection methods, we propose a training method that can jointly optimize the spectro-temporal filters and the neural net acoustic model built on them. In this special neural network achitecture, the filters are incorporated into the network and employed as the lowest layer of it. This allows us to tune the filters using backpropagation, and to manipulate them directly and not through their parameters. This method also has the advantage of reducing the task of filter set enhancement to that of a simple neural net training. Next, we show that we can enhance our manually selected filter set with this novel neural net architecture using the filter coefficients as initial values for the backpropagation training. The resulting filter sets were evaluated on the phone recognition task of the TIMIT corpus, using both clean and noise contaminated data; while cross-database phone recognition performance was evaluated on the "Szeged" Hungarian broadcast news database. The results we get demonstrate that the proposed filter optimization algorithm can outperform the usual feature selection-based methods, and that the filter set obtained by fine tuning the manual filters with the neural net algorithm performs even better, beating all the other methods in terms of performance.

Keywords: spectro-temporal features, Gabor filter, neural net, TIMIT

1. Introduction

In this paper, we investigate the selection and enhancement of a special class of spectro-temporal filters (Gabor filters) for automatic speech recognition purposes. Neurophysiological and biological studies have found that neurons responsive to spectro-temporal modulations play an important role in speech perception (Aertsen and Johannesma, 1981).

[†] The final publication is available at Springer via `http://dx.doi.org/10.1007/s10772-014-9246-4`

These findings motivated the researchers of Automatic Speech Recognition (ASR) to study the kind of preprocessing methods that extract spectro-temporal features. The most obvious approach is to apply the two-dimensional discrete cosine transform (2D DCT) on small spectro-temporal windows (Kovács and Tóth, 2010). This is a natural extension to cepstral processing, which applies 1D DCT on spectral vectors.

Visually inspecting the above-mentioned cortical neurons, one may find their response to be similar to that of the family of so-called Gabor filters, introduced by Gábor (1946). Indeed, it has been shown experimentally that these filters can be used to model the response profile of certain neurons (Jones and Palmer, 1987). These properties of Gabor filters made them a popular feature extraction method for various audio and visual classification experiments (Kleinschmidt, 2002a; Huang et al., 2005; Ezzat et al., 2007; Meyer and Kollmeier, 2008; Schädler et al., 2012).

There is a problem, however, that has to be addressed with each of these feature extraction methods: the selection of a reasonably small set of relevant features (i.e. filters) got from the huge variety allowed by the parametrization process itself. While in the case of 2D DCT there are some assumptions about which features should be kept (Ezzat et al., 2007), with Gabor filters we have less a priori information. This makes finding the optimal Gabor filter set quite difficult. Earlier on, most authors proposed applying automatic feature selection methods to find an appropriate set of filters (Kleinschmidt, 2002b; Ezzat et al., 2007; Meyer and Kollmeier, 2008). Unfortunately, these feature selection algorithms are very slow and are based on a greedy search that may yield suboptimal solutions. Although there are tasks where the greedy strategy may give acceptable results, in this paper we argue that this is not actually the case here.

We first demonstrate this by showing that a carefully designed, manually selected filter set can easily outperform the ones created by automatic filter selection methods. For this, we will first use the TIMIT phone recognition task with clean speech. One alleged advantage of spectro-temporal representations is that they process their spectro-temporal input locally, and so they are supposed to be more robust under noisy conditions. To test this, we will also compare the various algorithms on noise-contaminated versions of the TIMIT dataset. Neurophysiological studies suggest that a good set of spectro-temporal features should extract a set of invariant features that are independent of the actual training data. To see how well this assuption holds for our filters, we will also run cross-database evalutions on a Hungarian corpus that was recorded under quite different conditions. Then we perform several specially designed experiments that give an insight into what

makes the feature selection task defined here outstandingly difficult, and why it causes the simple greedy strategy to fail.

In Sect. 7, we will provide an alternative way of finding a proper set of spectro-temporal filters. This approach exploits the fact that the spectro-temporal filters and the subsequent neural net classifiers have very similar mathematical formulations. Consequently, the spectro-temporal filters can be interpreted as special types of neurons, and thus the backpropagation training of the network can be applied to them. This way the filter coefficients can be trained directly, which is much faster and more efficient than searching the parameter space that defines the shape of the filters. The results of our experiments clearly justify the superiority of this training method over the feature selection-based approach.

The backpropagation neural net training algorithm only guarantees that a local optimum will be found, and hence a suitable initialization of the parameters might help the training process. Below, we describe how we use the manually found Gabor filter set to initialize the neural net-based model, instead of using the standard random initialization approach. The results of our experiments indicate that in most cases this initialization can yield even better results.

2. Experimental setup

Before formalizing the spectro-temporal feature extraction method and the Gabor filters, we will describe the experimental tools used throughout this study.

2.1. Log mel-scaled spectrogram

All feature extracion methods discussed here process local patches of the log mel-scaled spectrogram. We computed the spectrograms using 400 samples (25 ms) per frame at 160 sample (10ms) hops, and applied a 1024-point FFT on the frames. These were then transformed to a log mel-scale of 26 spectral channels[1]. Each recording was normalized so as to give a zero mean and unit variance. Then, to avoid artificial down-weighting of low frequency bins, the lowest four channels of the spectrograms were mirrored.

[1] As some of our references (Kleinschmidt and Gelbart, 2002; Schädler et al., 2012) used log mel-scaled spectrograms with 23 channels, for purposes of comparison we ran the corresponding experiments using both 23 and 26 channels, and then in the subsequent experiments we used the configuration that performed the best.

2.2. Speech databases

All our experiments were conducted on the TIMIT speech corpus (Lamel et al., 1986) and the "Szeged" Hungarian broadcast news database (Gosztolya and Tóth, 2010). In the experiments performed on TIMIT, we followed the standard train-test partitioning of having 3,696 train sentences and a core test set of 192 sentences. The phonetic labels of the database were fused into 39 categories, as is standard practice (Lee and Hon, 1989).

The Hungarian speech database was the "Szeged" broadcast news corpus created at the MTA-SzTE Research Group on Artificial Intelligence and the University of Szeged. Recordings of seventy newscasts from eight television channels were cut into few sentence long blocks, and placed into one of three categories: noisy, spontaneous and clean speech. We used the last category, which contained well formed, articulated sentences with minimal background noise, typically uttered by news hosts in studio settings. Newscasts were partitioned into train, development and test sets: approximetely 5.5 hours were used for training, 2 hours for development and 1 hour for testing. All the recordings were orthographically typed, and the corresponding phonetic transcripts were created with a simple phonetic transcriber. The phonetic labels of the database were put into 52 categories.

To create a phone recognizer for the two tasks, we used an HMM/ANN hybrid model. In this scheme the frame-level phone posterior estimates of the neural net get combined by an HMM, derived using the Hidden Markov Model toolkit (Young et al., 2006) – which was modified in order to be able to work with neural net posteriors – and a simple bigram language model. The phone insertion penalty was tuned on the train set in the case of TIMIT, and on the development set in the case of the Hungarian database.

2.3. Adding noise

To evaluate the noise-robustness of feature sets, we artificially contaminated the core test set of TIMIT with noise. To this end, we applied the FaNT tool (Hirsch, 2010) to add the noise with the proper signal to noise ratio (20 and 10 decibels). This is similar to how the noisy dataset of Aurora-2 was created from the original TIDigits corpus, and the noisy dataset of Aurora-4 was created from the clean Wall Street Journal database. We used two types of noise samples taken from the NOISEX-92 database (Varga and Steeneken, 1993). The first noise type was pink noise, which has the highest energy at 0Hz and tails off at higher frequencies. The second noise type was babble noise, which simulates the effect of people talking in the background. As a

third type of noise, we created a bandlimited noise sample by filtering white noise with a bandpass filter, with a passband between 3,000 and 5,000 Hz.

2.4. Student's t-test

The training of neural nets starts from a random initialization, and so training the same network on the same data may give slightly varying results. To decrease this variance, each neural net training experiment was repeated ten times, and we report the average of the resulting ten accuracy scores. To decide whether the difference between the results of two feature sets was significant, a two-tailed Student's t-test was applied, with unequal variance. We considered the difference significant if the p value resulting from the t-test was smaller than 0.05.

3. Spectro-temporal filters

Let us now formalize the process of spectro-temporal feature extraction. This approach takes spectro-temporally localized patches from the spectrogram of the speech signal, and creates features for ASR purposes by processing them using standard filtering methods. Such a spectro-temporal feature o can be got by applying the formula

$$o = \sum_{f=0}^{N} \sum_{t=0}^{M} P(f,t)F(f,t), \tag{1}$$

where N and M are the height and width of patch P and filter F, which must have the same size. One can get a set of features by using several filters with different coefficients, and/or by applying them at different positions along the time-frequency plane.

As can be seen, the calculation of the spectro-temporal features is simple; the real question is how to get proper coefficients for the filter $F(f,t)$. We discuss several methods for this in the following.

3.1. 2D DCT

The well-known mel-frequency cepstral coefficient (MFCC) feature extraction method processes mel-scaled spectral vectors by applying DCT on them. This gave us the idea of processing the spectro-temporal patches via 2D DCT. This corresponds to performing the filtering defined by (1) with the following filter coefficients:

$$F_{pq}(f,t) = \cos\frac{\pi \cdot f \cdot p}{N} \cos\frac{\pi \cdot t \cdot q}{M}, \quad \begin{matrix} 0 \le q \le N-1 \\ 0 \le p \le M-1 \end{matrix} \tag{2}$$

where N and M are the respective height and width of the filters for f and t, while p and q specify the modulation frequencies of the filter along the frequency and time axis.

By definition, for a patch of size $N \times M$, a 2D DCT returns the same number of coefficients. We have several results indicating that these are not equally important for representing the underlying acoustic content. As regards the frequency axis, the good performance of MFCC clearly shows that it is sufficient to keep the low-order coefficents. We have several similar results regarding time-domain modulations (Kanedera et al., 1999). Using these rules as heuristics, Kovács and Tóth have obtained good results with 2D DCT modulation features, by keeping just the lowest order 9 coefficients (Kovács and Tóth, 2010; 2011).

3.2. Gabor filters

Using a DCT for the processing of spectral patches is an engineering solution that does not have any biological foundation. However, we have now a better understanding of how the human brain processes speech signals, and the response of the spectro-temporal receptive fields found in neurophysiologial studies has similarities with the response of the so-called Gabor filters. This gave us the idea of using Gabor filters as an engineering model of these receptive fields.

While Gabor filters are commonly defined as a product of a complex sinusoid carrier and an envelope function, the exact definition of Gabor filters may vary slightly from author to author. Ezzat et al. (2007) defines Gabor filters as a product of a two-dimensional Gaussian (3) and an oriented sinusoid (4). That is,

$$W(f,t) = \frac{1}{2\pi\sigma_f\sigma_t} e^{-\frac{1}{2}\left(\frac{(f-f_0)^2}{\sigma_f^2}+\frac{(t-t_0)^2}{\sigma_t^2}\right)} \tag{3}$$

$$S_{\Omega,\omega}(f,t) = e^{i2\pi\left(\frac{\Omega}{N}f+\frac{\omega}{M}t\right)}, \tag{4}$$

where f and t iterate over the frequency and time span of the window, and σ_f^2, σ_t^2 specify their respective bandwiths. N and M specify the transform size, while Ω and ω specify the slanting of the sinusoid as well as its periodicity. If we use just the real part of the Gabor filter, Eq. (4) can be rewritten in the following form:

$$S_{\Omega,\omega}(f,t) = cos\left(\frac{\pi \cdot f \cdot 2\Omega}{N} + \frac{\pi \cdot t \cdot 2\omega}{M}\right). \tag{5}$$

Note that if either p or q in (2) equals zero, and we choose the following assignments for (5): $\Omega = \frac{p}{2}$, $\omega = \frac{q}{2}$, then equations (2) and (5) are exactly the same. Later on, this similarity will be exploited.

As can be seen, with the above formulas we can control the shape of the filters via the parameters Ω and ω. However, it is not clear how many filters should be used and how their parameter values should be chosen for optimal speech recognition performance. Below, we shall describe several ways of finding a good parameter set for speech recognition.

4. Finding an optimal set of Gabor filters by feature selection methods

Even if we fix the filter size parameters N and M of (4), the parameters Ω and ω are continuous and thus define an infinite space of possible filters. Our goal here is to select a reasonably small number (i.e. at most a couple of hundred) from among these filters in such a way that the selected filters provide useful features for speech recognition purposes. The methods discussed here all seek to solve this problem by means of feature selection methods. That is, all the filters of the search space are systematically evaluated, resulting in a huge feature space of possible acoustic features. Then, by selecting those features that yield a good speech recognition result, we also inevitably select the corresponding Gabor filters. Of course, the big question here is to find a proper feature selection method that chooses the best candidates from the huge space of Gabor filters. Hence, a major part of research in Gabor filters is devoted to this filter selection task. Here, several points need to be considered. Namely,

1. We would like to create a simply parametrized filter family that detects all relevant speech phenomena, and which is hopefully independent of the actual training database. That is, although the filter set will be optimized on a given speech corpus, we would prefer to get a filter set that gives a nice recognition performance on different databases as well.

2. Given the variety of spectral and temporal properties of human speech recognition, it is expected that a huge search space needs to be examined. As we intend to measure the quality of the feature set candidates by evaluating their usefulness in speech processing, our feature selecton algorithms will necesarrily have long runtimes. Also, it should be mentioned that a global optimum could be guaranteed only by evaluating all the feature subsets. This is clearly impossible in this case, so we must employ some very good heuristics during the search.

3. These filters do not constitute an orthogonal representation of the signal and lead to highly correlated features. The correlation depends on the similarity of the filter parameters, but it is hard to quantify. We can expect that their correlations will make the search task even more difficult.

Reviewing the literature of Gabor filters, we found that most authors apply automatic feature selection methods to find a proper subset of filters. These automatic methods are all built on machine learning principles (Tasi, 2001; Kleinschmidt, 2002a; Kleinschmidt, 2002b; Sun et al., 2003; Meyer and Kollmeier, 2008), with the most popular algorithm being the Feature Finding Neural Net (FNNN) (Gramms, 1991). In the following, we will briefly present and evaluate this algorithm, along with another, general-purpose feature selection method called Sequential Forward Floating Selection (SFFS) (Somol et al., 2010). Then we will manually select a small set of filters, where the manual selection is guided by some simple heuristics. We will show that this manually selected filter set almost always outperforms the ones found with automatic methods, which clearly demonstrates the weakness of the automatic selection algorithms. We will state several arguments and also describe the results of some experiments in order to explain why the feature selection methods seem to fail on this task.

4.1. Automatic feature selection algorithms

To evaluate the performance of these methods, we created filter sets with two automatic feature selection algorithms. In the following, we will briefly introduce these algorithms. As a detailed discussion of these methods is beyond the scope of our paper, we kindly refer the reader to Gramms (1991) and Somol et al. (2010).

We selected the size and overlap of our filters based on the literature and also on experimental findings. We set the analysis window length to 100 ms, based on studies of human speech understanding (Tiitinen et al., 2012). In the experiments, we also applied the standard Δ and $\Delta\Delta$ features in order to incorporate more temporal information. The window length combined with our aim of creating square filters also defined the height of our filters. Next, we restricted the overlapping of filters along the frequency axis to between 50% and 60%. In Sec. 4.2, we used the same settings for the filter sets with manual methods.

After setting the filter size and overlap, in order to make the definition of the search space simpler, we slightly modified the original formulation of the Gabor filters given earlier. Our new formula will be

$$S_{\Omega,\omega}(f,t) = e^{i2\pi\left(\omega_f \cdot f + \omega_t \cdot t\right)}. \tag{6}$$

This re-formulation is motivated by the observation that although the sinusoid (4) of a Gabor filter is defined by four variables, its output does not depend on the individual values of Ω and N, but rather on their ratios (see above). This also holds for ω and M. Thus we may introduce $\omega_f = \frac{\Omega}{N}$ and $\omega_t = \frac{\omega}{M}$, turning (4) into a two-parameter formula.

The next step required by the filter selection algorithms was to define the interval and resolution of ω_f and ω_t. To keep the periodicity of the sinusoids within a reasonable interval, the minimum and maximum parameter values were set to -0.14 and 0.14, respectively, and we chose to discard similar filters by setting the resolution of the parameters to 0.004 (for symmetry reasons, ω_f was run from 0). Lastly, for computational reasons we only kept the real part of Gabor filters. Even with the above-mentioned restrictions, with the width of the Gaussian limited to one third of the patch size and its peak fixed to the patch centre (f_0, t_0), ω_f and ω_t still define a huge parameter space.

4.1.1. *Feature Finding Neural Nets (FFNN)*

The Feature Finding Neural Net (Gramms, 1991) algorithm was popularized for the purpose of Gabor filter selection by Kleinschmidt and Gelbart (2002). During its operation, this algorithm always has a current set of candidate filters. This candidate filter set consists of $D + 1$ filters, where $D = 9$ is the number of filters we would like to have in our final set as 9 filters proved to be sufficient in previous studies (Kovács and Tóth, 2010; 2011; 2013). Hence, first of all we have to to initialize the algorithm by creating an initial set. We do this simply by randomly selecting $D + 1$ filters. Then the algorithm consists of repeating two steps. Namely,

1. Evaluate each D-element subset of the current candidate set. This is performed by training a two-layered neural net on 90% of the training data using the remaining 10% of it as the stopping criterion. Then this latter 10% of the training data is used as a validation set for evaluating the performance of the subsets.

2. Take the subset that performed best on the validation set, and from the candidate set eliminate the filter that was not in the best performing subset. Choose a filter randomly as a replacement.

By repeating these steps, we expect to gradually improve our filter set. The algorithm terminates when it attains a local minimum and cannot improve the feature set any further.

Since the resulting set depends on the initial set, in the experiments we performed the computations fifty times using fifty different initial random sets. We trained thirty three-layered neural nets on each resulting set, and then chose the best performing one by comparing the

average accuracy values on the validation set. The set of filters resulting from this process will be referred to as the *FFNN set*.

4.1.2. *Sequential Forward Floating Selection (SFFS)*
To find a good set of Gabor filters, we also experimented with another, general-purpose feature selection method called Sequential Forward Floating Selection (SFFS). A detailed analysis of this algorithm, was given by Somol et al. (2010). This method consists of the following main steps:

1. Start with an empty set of filters.
2. Expand the filter set by finding and adding the filter that improves classification accuracy the most. This is done by adding one filter at a time, retraining the classifier on 90% of the training data, and evaluating it on the validation set (the remaining 10%).
3. Find and discard that filter whose absence has the least detrimental effect on the classification performance. This is done by removing one filter at a time, and retraining the classifier on the reduced set. If the new score is even better than the previous best one (with the same cardinality), then repeat this step. Else go to the second step.

We repeated the feature selection experiment using this algorithm, and the resulting set of 9 Gabor filters will be referred to as the *SFFS set*.

4.1.3. *Preexisting[2] Gabor filter sets*
Kleinschmidt and Gelbart (2002) not only described the FFNN algorithm in their paper, but also made the three filter sets derived from their experiments publicly available. These filter sets are referred to as *G1, G2, G3*, and are available at the Berkeley website (Gelbart et al., 2013). Each set consists of 60 filters, among them there were real, imaginary and magnitude responses of Gabor filters, as well as purely spectral, purely temporal and spectro-temporal filters. The parameters of G1 and G2 were trained on TIMIT, while the parameters of G3 were optimized on the 'Zifkom' corpus of German digits.

Schädler, Meyer and Kollmeier in their paper (2012) applied a different approach for selecting their Gabor filter set, and their method has some similarities with our manual method (see Sect. 4.2). They chose the parameters of spectral modulation frequencies based on MFCC and temporal modulation frequencies based on RASTA processing. Then they carried out further optimization of the parameters by performing some ASR experiments on the Aurora-2 task, creating a filter bank of

[2] Here, preexisting means that these filters were taken 'as is' from other authors.

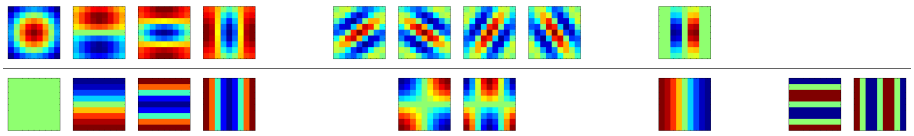

Figure 1. Manual set of Gabor filters (first row) and 2D DCT (second row) filters, with the corresponding filters vertically aligned to emphasize the similarities. The patch size is 9x9.

41 filters. Next, by subsampling the filter output for each filter, they selected the representative filter channels, which was again carried out on the Aurora-2 database. The result of this process was a filter set containing 311 elements in the case of the 23 channel log mel-scale spectrogram, and a filter set containing 356 elements in the case of the 26 channel log mel-cale spectrogram. The resulting filter sets and the code for feature extraction are available at the website of Oldenbury University (Ossietzky, 2013) These filters will be referred to here as the *SMK set*.

4.2. Manually selected filter set

Apart from the automatic feature selection algorithms presented above, we also created filter sets manually. For this manual selection process we used two sorts of a priori information. First, as in earlier studies it has been found that processing the spectro-temporal patches via 2D DCT yields good results (Kovács and Tóth, 2010), we conjectured that the filters defined by the 2D DCT coefficients would serve as a good starting point for our search. Second, a visual inspection of the shape of the most frequent transition types in spectrograms also served as a heuristic concerning which filters ought to be included in the final set. In the following, we will elaborate on how we arrived at the proposed set.

In the paper of Kovács and Tóth (2011), the local patches were processed by applying a 2D DCT. Retaining nine coefficients proved to be an efficient representation in this paper, which motivated us to take advantage of the similarity between 2D DCT and the sinusoid form in the Gabor filter, as outlined in Sect. 3.2. This similarity guided us in the selection of our first four Gabor filters. The first row of Fig. 1 shows how the selected Gabor filters approximate the corresponding 2D DCT filters. As can be seen, we use a rotationally symmetric filter to detect the energy of the processed patch, while with the use of the three other filters we seek to capture the change in frequency and time, respectively.

Conventional MFCC features extended with the delta and delta-delta coefficients capture transitions in time and frequency independently. In contrast, with 2D DCT and Gabor filters, we can extract slanted energy transitions in the time-frequency domain in an explicit way. Presumably this property is the key advantage over conventional features because it encodes spectro-temporal representations directly. This motivated the selection of our second group of filters. The second four (slanted) Gabor filters (shown in the first row of the fifth to eighth columns of Fig. 1) were selected by distributing the "slanting" of the filters uniformly. With these filters we intend to detect spectro-temporal phenomena like formant transitions.

Next, we noted the similarity between the seventh filter (in row two) and the delta vector. Motivated by this, we tried to construct the last Gabor filter in such a way that it approximated the computation of the delta vector. As our filter set relies heavily on an earlier 2D DCT set, the cardinality of our filter set was also chosen to be nine. Below, we will refer to the set of Gabor filters shown in Fig. 1 as the *Manual set*.

4.3. Randomly selected filter sets

Besides the automatic and manual filter selection methods, we also created ten additional filter sets by simply choosing filters randomly. These filter sets will serve as a baseline during the evaluation of the various filter selection algorithms. We will refer to the average performance of the ten random filter sets as *Random avg*. We will also list the performance score of the best performing random filter set that was chosen based on the results obtained on the validation set of TIMIT, which will be referred in the tables as *Random best*.

5. Experimental results and discussion

Now we will describe the experimental settings used when we carried out our comparative evaluation of automatic and manual Gabor filter selection methods, and list the results for each testing scenario.

5.1. Representation of frames

Our frame level feature vectors were created from the local patches as follows. First, each filter in the set was evaluated on each patch of the spectrogram. In the case of the filter sets we created using automatic methods or manually, the patches had a length of 9 frames and a height of 9 channels, with a step size of 4 mels (4 channels) in frequency. In the case of the pre-calculated filter sets of Kleinschmidt and Gelbart (2002),

the sets were defined so that the filters covered the whole frequency range, hence no frequency-domain step size was required. Also, the filter set of Schädler et al. (2012) did not have a unified step size either, because the filters were selected uniquely by evaluating each filter on each frequency band, without taking into account their overlap with the neighbouring filters.

Having evaluated the filters, the feature values obtained were associated with the centre position of the patch, thus giving a set of feature values for each time position. This set of features were used to classify the given frame, as will be described next. As is usual with MFCC, in order to incorporate more temporal information we added the Δ and $\Delta\Delta$ (Δs) features to the feature sets, and used 9 neighbouring frames of feature vectors to train the neural net classifiers. The only exception was the case of the *SMK set*. Based on the results of some pilot experiments and because of the higher dimensionality of the frame level feature vectors produced by this set, we eventually decided not to add the Δ and $\Delta\Delta$ features, and we used only 5 neighbouring frames during neural net training.

5.2. Neural network classifier

In each experiment, the classifier applied was a multilayer neural network with a hidden layer of 1,000 neurons during filter set construction and 4,000 neurons during the classification experiments. (We only used two layers in the neural nets of the FFNN algorithm because the number of nets to be trained would have produced a huge computational cost.) The output layer applied the softmax nonlinearity, while the hidden neurons worked with the sigmoid function. The number of output neurons was set to the number of classes (39 in experiments with TIMIT, and 52 in experiments with the Hungarian speech database). Naturally, the number of inputs varied based on the number of features extracted by the actual filter set, whose difference might be compensated for by the relatively large size of the hidden layer. The neural net was trained with random initial weights using standard backpropagation on 90% of the training data in semi-batch mode, and the remaining randomly selected 10% of the data set was used as the validation data set. As we explained in Sect. 2.4, for each feature set ten independent neural nets were trained, and the average of the results got from applying these neural nets on the test set was reported.

5.3. The clean speech experiments on TIMIT

Table I lists the phone recognition accuracy scores we got from applying our method on the clean TIMIT data with the various feature sets. As a

Table I. Phone recognition accuracy scores got on clean core test set of TIMIT. The best result, and the results not significantly different from it are shown in bold.

Feature set	No. of mel channels	No. of features	Phone accuracy
MFCC + Δs	26	39	72.95%
SFFS set + Δs	26	162	**73.15%**
FFNN set + Δs	26	162	**73.18%**
G1 + Δs	26	180	72.57%
G2 + Δs	26	180	72.20%
G3 + Δs	23	180	64.73%
SMK set	23	311	71.51%
Manual set + Δs	26	162	**73.22%**
Random avg + Δs	26	162	72.34%
Random best + Δs	26	162	**73.14%**

baseline, the recognition results got with the standard MFCC features are shown in the first line of the table. The next two rows show the results of the two automatic filter selection methods. Both of these slightly outperformed MFCC, giving us the first indication that the automatic selection methods work as they should.

The table also contains the results got with the preexisting *G1-3* sets. These all gave significantly worse results than either MFCC or the filter sets we got by using automatic feature selection methods. In particular, the performance of set *G3* is strikingly low, compared to the two other sets. This result seems strange to us as in the original paper (Kleinschmidt and Gelbart, 2002) *G3* was supposed to have been the best performing set. A possible explanation for this phenomenon might be the fact that while the parameters of *G1* and *G2* were optimized on TIMIT, *G3* was optimized on the 'Zifkom' corpus of German digits. This suggests that filter sets were overtrained on the particular database, either overlearning on language specific information (German vs. English) or overlearning on the acoustic properties of the database.

Lastly, Table I shows the results obtained with the manually and randomly selected feature sets. As can be seen, the *Manual set* slightly outperforms every other feature set, and the difference is significant in the case of all the predefined Gabor filter sets as well as in the case of the baseline MFCCs. This strongly suggests that the automatic filter

selection methods fail to find the optimal features, as the optimal set should be at least as good or better than the manually found set.

We observe even more strange results when we examine the performance of the randomly selected feature sets. It is really surprising how close the average of the random results is to the performance of carefully designed selection methods. In fact, the average random score is significantly better than that of two given Gabor filter sets (*G3* and *SMK*), while only one of them (*G1*) produces a significantly better result. Furthermore, the performance of the best random set is not significantly different from that of the best optimized feature set. We think this behaviour is rather unexpected and it probably explains why the simple greedy feature selection strategy fails. We shall discuss these issues later in detail in Sect. 6.

5.4. Noise contaiminated speech experiments on TIMIT

While all the components of the MFCC feature vector are extracted from the full spectrum, the spectro-temporal filters are bandlimited. Thus, any type of noise that does not cover the full spectral range should contaminate only a subset of our spectro-temporal features – in constrast to the standard cepstral features. This is why we expect spectro-temporal features to be more robust to noise. Below we test this assumption by evaluating our Gabor filter sets and the neural network classifier on the noise-contaminated core test set of TIMIT. We should emphasize here that the filter selection and classifier training steps were NOT repeated. That is, these experiments use the same features and neural network parameters that were found to be optimal in the previous set of experiments *on clean data*.

The results got with various types of noise added can be seen in Table II. In general, the spectro-temporal filters proved much better than MFCCs when babble or pink noise was added in almost every case, independent of the filter selection method applied. However, only the best selection method (*Manual*) gave better results than the MFCCs in the case of bandlimited noise. A thorough analysis revealed that this relative failure of Gabor filters for bandlimited noise can be attributed to the simple normalization technique we used: because of the narrow and very loud noisy spectral channel the remaining clean channels are also suppressed during normalization. In the future, we need to think of a better normalization technique to alleviate this effect.

Comparing the two filter sets we got using automatic feature selection methods (*SFFS* and *FFNN* set), we see that while under clean conditions there was no significant difference between their performance scores, which does not hold in the case of noise: the *SFFS set* signifi-

Table II. Phone recognition accuracy scores got on the core test set of TIMIT, contaminated with different types of noise. The best result in each column (and those results significantly not different from it) are shown in bold.

Feature set	Babble		Pink		Bandlimited	
	20db	10db	20db	10db	20db	10db
MFCC + Δs	57.43%	35.32%	49.93%	30.05%	61.78%	**50.48%**
SFFS set + Δs	**63.10%**	**47.08%**	55.77%	34.18%	60.59%	47.40%
FFNN set + Δs	62.75%	46.02%	55.24%	33.18%	58.50%	46.97%
G1 + Δs	59.77%	39.59%	**56.20%**	**35.99%**	54.32%	44.32%
G2 + Δs	60.75%	40.52%	55.86%	**35.92%**	51.15%	41.91%
G3 + Δs	51.97%	37.49%	51.05%	**35.88%**	39.97%	30.85%
SMK set	54.28%	29.97%	55.14%	33.77%	54.04%	44.73%
Manual set + Δs	**63.21%**	**47.03%**	55.74%	34.24%	**63.59%**	**50.87%**
Random avg + Δs	62.51%	45.99%	54.26%	33.42%	60.25%	47.64%
Random best + Δs	62.87%	**46.87%**	55.05%	33.77%	60.06%	46.93%

cantly outperforms the *FFNN* set in all but one case. Now, when we proceed to evaluate the manually selected filter set we observe that the *Manual set* not only outperforms the baseline MFCC in every case, but it also significantly outperforms the *FFNN* set in all cases, and it is never significantly worse than the *SFFS* set. Extending the comparison to the preexisting filter sets, we see that in the case of pink noise the *G1* and *G2* sets perform slightly better than the manual sets, but in every other case the performance score of *Manual set* is significantly better than the performance score of any preexisting Gabor filter set. Overall, we can say that the *Manual set* performed better in noisy environments than any other feature set we examined. This suggests that the simple heuristic rules we applied during manual selection (e.g. to cover the most frequent modulation types) works better in noisy conditions than the heuristics applied in the SFFS and FFNN feature selection methods. The fact that performance of *G3* is again noticably worse than the performance of *G1* and *G2* reinforces our belief about the cross-database performance problems of the automatic selection methods (remember that among *G1-3* only the parameters of *G3* were trained on a speech database different from TIMIT).

Now, when we turn our attention to the *random* sets, we see that the set that was selected as 'best' based on its performance on the clean validation set performs better than the average in most cases,

though not under bandlimited noise conditions (as was explained earlier, the poor performance under bandlimited noise is caused by the weakness of the normalization technique). In general, the random filter sets outperform the predefined sets under babble and bandlimited noise conditions, while the predefined sets work much better in the case of pink noise. But, the most persuasive thing for us is that the performance gap between the random sets and the best set is always very small, i.e. just 1-2%. We expected a strong method to have a consistently better performance, but this wasn't apparent here.

5.5. Clean speech cross-database experiments on the Hungarian speech database

Now we will evaluate our filter sets on the Hungarian "Szeged" broadcast news corpus. As in the case of the noisy TIMIT tests, we did not repeat the selection of the filters, but used the filter set optimized on the clean TIMIT. These cross-language and cross-database tests were motivated by theoretical and practical aspects. Theoretically, one would expect the cortical receptive fields – and thus the Gabor filters that model them – to extract some sort of invariant features that should not depend on a training database[3]. And a practical reason is that the automatic feature selecton methods like *FFNN* and *SFFS* are extremely slow, so it would be advantageous if the feature selection process did not have to be repeated for each training corpus.

The phone recognition results got on the Hungarian corpus are shown in Table III. Evidently, the MFCC features yielded the best results, and from among the various Gabor filter sets only the *Manual set* managed to come close. All the feature selection algorithms and the predefined filters sets produced much worse scores. This point clearly shows that the heuristics applied in the manual selection process are more general than the heuristics of the database-driven feature selection methods. By comparing the results got using various filter selection algorithms with each other, we see that the feature set created by SFFS significantly outperformed its FFNN created counterpart. As we observed a similar tendency with TIMIT, SFFS here seems to be a better selection algorithm than FFNN. However, the resulting filter set still performs significantly worse than MFCCs, so unfortunately our hope that the filter optimization would not have to be repeated for each training database did not materialize. Also, we can see that the best randomly selected filter set gave scores almost as good as that

[3] One might argue that these features may be different for different languages, however. This issue needs to be examined to see if it really the case

Table III. Phone recognition accuracy scores got on the clean test set of the Hungarian speech database

Feature set	Phone recognition
MFCC + Δs	**74.97%**
SFFS set + Δs	73.94%
FFNN set + Δs	73.51%
G1 + Δs	74.09%
G2 + Δs	72.84%
G3 + Δs	63.68%
SMK set	73.05%
Manual set + Δs	74.67%
Random avg + Δs	72.79%
Random best + Δs	73.63%

of SFSS, again suggesting that the filter selection algorithms fail to achieve their goal.

Examining the available Gabor filter sets, we notice similar patterns as in the case of TIMIT: while the *SMK* set performed in the mid-range, *G3* yielded much worse results than those got with the other feature sets.

6. What is wrong with the automatic selection methods?

During the experiments we repeatedly found that despite the time invested in the elaborate feature selection algorithms, the filter sets they produce were easily outperformed by a simple manually selected set, and even a randomly selected set could yield a very similar performance. This is surely not what one would expect. Hence below we try to provide an insight into the reasons for this.

Feature selection is a very difficult task when the features are strongly correlated, which is clearly the case with Gabor filters. It is hard to tell how the overlapping information content of the features assist each other, and it is also impossible to tell in advance how the correlations influence the performance of such a complex classifier as a multilayer neural network. Hence, the only way of finding a truly globally optimal

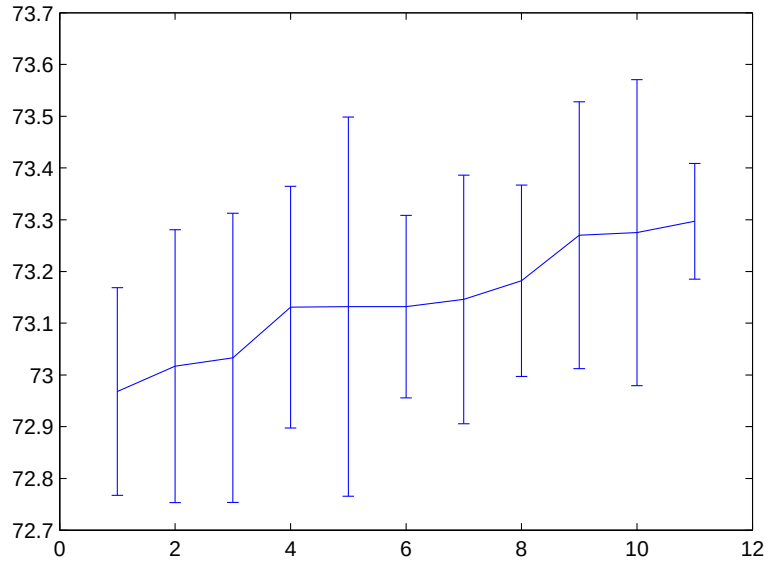

Figure 2. The phone recognition accuracy scores got on TIMIT core test set after performing a random filter replacement on the *SFFS set*. The error bars correspond to the scatter of the scores obtained when we repeated the training ten times.

feature subset would be to evaluate each possible subset. This is clearly impossible, as the number of subsets grows exponentially with the number of features. The need to reduce computational costs to a managable level encouraged the researchers to use the the greedy search techniques such as the FFNN and the SFFS algorithms. Even when done this way, the given algorithms are very slow (they required several days to finish on TIMIT, which is now regarded as a very small corpus). Both of these algorithms rely on the heuristics that the evaluation of each subset can be reduced to a search that adds and removes only one filter at a time. Even though the search strategy allows backtracking (which makes it more flexible), convergence would require the replacement of a feature to significantly change the actual performance of the hypothesis set. In our case, however, replacing one filter in the set with another one in most cases has only a negligible effect on the classification score, and this causes the algorithm to oscillate. We designed a special experiment to demonstrate this behaviour. We took the best filter set found by SFFS and replaced the "first" filter (i.e. the filter that was chosen first by the SFFS method) with ten arbitrary Gabor filters, resulting in ten additional filter sets. As this filter was chosen first because it

was the best "lone" filter for separating the classes, it is reasonable to expect that its replacement should have the biggest impact on the phone recognition scores. To evaluate these ten filter sets, we trained ten independent neural nets on each of them. The results got on the TIMIT core test set are shown in Fig. 2, with the score of the original *SFFS set* being in the *8th* column. We observed that it produced no significant difference between the 73.15% phone recognition accuracy score of the original *SFFS set* and the average phone recognition accuracy score of its derivatives (i.e the average performance of the sets created from the original *SFFS set* by replacing one filter with an arbitrary new filter), which is 73.14%. It is also true that the recognition accuracy scores of all the individual SFFS derivatives are very similar to the phone recognition accuracy of the original *SFFS set*. In fact, the score of the original *SFFS set* is not even the best (the best result being 73.30%).

As in all our experiments, the training of the neural network on each set was repeated ten times. Also, we explained in Sect. 2.4 that neural network training algorithm used random initial parameter values, and because of this the resulting scores had a small scatter. This was plotted in Fig. 2 using the usual error bars. Clearly, the scatter caused by the random factor of the training process is larger than the scatter caused by the random replacement of the first filter. Hence, the results of this experiment demonstrates that, in the general case, randomly replacing one filter in the set by another one brings about such a small improvement that it can be easily swamped by the "noise" of the ANN used to evaluate the given set. If there are no filters that clearly stand out from the rest, then the simple selection strategy used by our algorithms is doomed to oscillate without convergence. The fact that randomly selected filter sets yield such good results strengthens this conclusion. It suggests that there is a very small performance gap between a randomly selected set and the optimal one, and there are no clearly "much better" and "much worse" filters. This makes the task of an algorithm that decides on each feature locally rather difficult. A possible improvement would be to modify the selection algorithms so that they restrict the addition of a filter to a minimum gain in classification accuracy, or to a certain level of dissimilarility from the previously selected filters.

The reader may find the notion of the good performance of randomly selected features rather surprising and counter-intuitive. However, lots of studies on various real-life machine learning tasks find that creating a representation using a large set of overcomplete random basis functions is just as good as putting a lot of effort into carefully designing a small, orthogonal set of basis functions. The application of sparse, overcomplete bases was first proposed in image recognition, based on

observations with vision (Jones and Palmer, 1987), but quite recently similar studies in speech recognition were also carried out. For example, Vinyals and Deng (2012) found that representing the speech signals by means of a randomly selected vector set gave phone recognition results that are just as good as those using an optimized, orthogonal basis vector set.

Further evidence on the applicability of random representations is given by the success of the Extreme Learning Machine (ELM) (Huang et al., 2006). The extreme learning machine is virtually a neural network with two hidden layers. During training the lowest layer, as usual, is initialized in a random way, but then *it is not trained at all*, and only the weights of the upper layer are optimized. This strategy seems to make no sense at first, but in fact it can give very good results, and in most cases it proves no significantly worse than the more tedious training of the full network. The extreme learning machine can be interpreted as if the first layer represented the input by means of a large set of random basis vectors, and then the second layer performed learning over this special representation. The success of this algorithm in practice also reinforces the view that there is only a very small performance gap between a randomly chosen feature set and the optimal one, and hence *this optimization is a very difficult task that cannot be solved by such naive strategies as the ones used by SFFS and FFNN.*

7. The joint optimization of spectro-temporal features and neural net classifiers

As we saw above, the feature selection algorithms such as FFNN and SFFS are terribly slow and fail to find an optimal feature set. Speed would not be that important if they could produce a feature set that would fit other data sets as well. However, the experiments did not demonstrate this data-independence property in filter sets created using automatic feature selection methods. In fact, the manually selected filter set gave better results in the cross-database tests. Unfortunately, the selection heuristics provide no guarantee that the set selected manually is optimal, and the manual selection process does not offer the chance to improve the filter set when more training data becomes available. Thus, we think that the filter set should be chosen automatically, but with a better algorithm than the ones presented so far. In the following, we propose a method that can jointly optimize the spectro-temporal filter parameter values and the weights of the neural net classifier.

In traditional speech recognition, the feature extraction and the classification (likelihood estimation) steps are conventionally performed

in two distinct steps. Although there are exceptions that try to incorporate the feature extraction parameters into the optimization of the acoustic models, such studies are quite rare (Biem et al., 1995; Lee et al., 2001; 2003). More recently, with the invention of deep neural nets, several authors suggested that these new type of networks are powerful enough to accept a less well prepared input than is usual with standard HMMs. For example, it now seems widely accepted that deep networks work more efficiently on a simple mel-spectral representation than with MFCCs (Mohamed et al., 2012). Some authors even tried to train deep networks on raw acoustic data (Jaitly and Hinton, 2011; Palaz et al., 2013). In a recent study, the training of a convolutional neural net was extended to the optimization of the feature extraction filter bank (Sainath et al., 2013).

The solution we propose here is based on a similar concept, but we use spectro-temporal filters instead of a conventional spectral filter bank. This method treats the feature extraction filters as the lowest layer of a neural net, and lets the training algorithm tune the filter coefficients as well. In the following we explain why and how the spectro-temporal filters can actually be treated as special type of neurons in a neural network.[4]

To explain how this approach works, let us examine the operation of a simple perceptron model. In general, the output of a single neuron can be obtained using the formula

$$o = a\left(\sum_{i=1}^{L} x_i \cdot w_i + b\right), \tag{7}$$

where $\mathbf{x}$ is the input of the neuron, L is the length of the input, $\mathbf{w}$ is the weight vector, and b is a bias corresponding to that neuron. For the activation function a we usually apply the sigmoid function; but it is also possible to create a linear neuron by setting a to the identity function. In that case, setting $b = 0$ and $L = N \cdot M$, and representing filter F and patch P of (1) in vector form, (which is actually just a notational change), namely

$$o = \sum_{f=0}^{N}\sum_{t=0}^{M} P(f,t)F(f,t), \tag{1}$$

we see that (1) in Sec. 3 is just a special case of (7). This means that the spectro-temporal filters can be integrated into an ANN classifier

[4] The basic idea of the algorithm has been introduced in a conference paper (Kovács and Tóth, 2013). Here, we present several refinements and a more thorough evaluation of this approach.

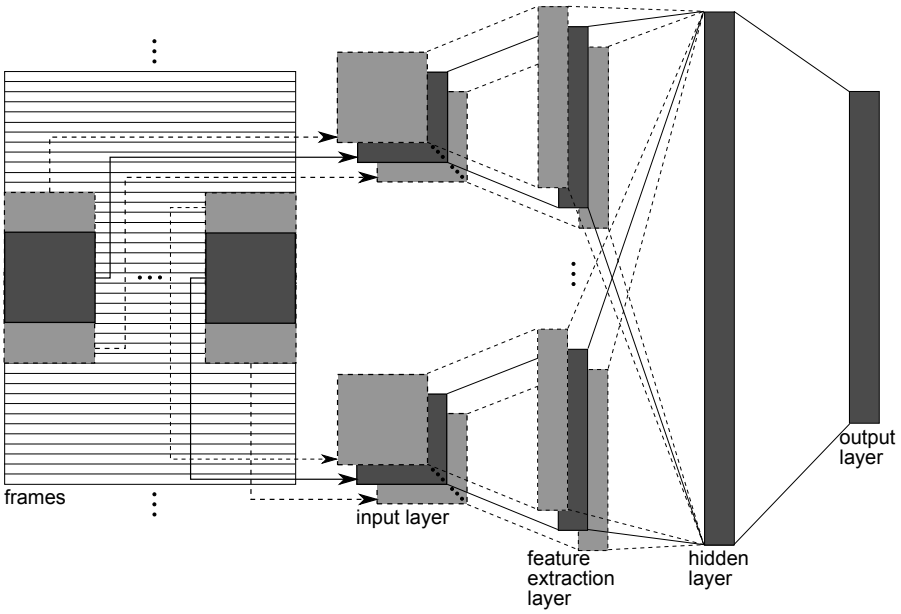

Figure 3. Structure of the ANN for joint feature extraction and classification. The boxes in light grey correspond to convolutional units that have shared weights, with the corresponding unit highlighted in a darker shade of grey.

system as special neurons, with the filter coefficients being the actual weights of the given neuron.

7.1. Structure of the ANN for combined feature extraction and posterior estimation

Figure 3 shows the proposed ANN structure that can perform spectro-temporal feature extraction and classification (phone posterior estimation) in one step. When compared to a conventional neural net, the main difference is the introduction of a feature extraction layer. The dark grey areas placed on the spectral frames in Fig. 3 show how the spectro-temporal patches of the speech signal are extracted to form the input data for the input layer. Then the linear neurons in the feature extraction layer perform the spectro-temporal filtering of (1). The output of this layer is channelled into the hidden layer, and from this point on the system behaves just like a conventional neural net. Notice that this structure can also operate along the lines of the classic scheme where the feature extraction and the classification steps are separate. For example, if the weights of the feature extraction layer were initialized with Gabor filter coefficients and were left unaltered during training, then the model would be equivalent to a more traditional

system, and the incorporation of the feature extraction step into the system would be just an implementational detail. The real novelty here is that we extend the scope of the training to the feature extraction units as well.

It is well known that integrating a longer temporal context into the acoustic features can significantly improve recognition performance. In HMM-based recognition the Δ and $\Delta\Delta$ features are used for this purpose, while in HMM/ANN hybrids a common technique is to feed several neighbouring acoustic vectors into the neural net (Bourlard and Morgan, 1994). Although spectro-temporal features process longer time intervals than tradional techniques (such as MFCC), Kovács and Tóth have observed that adding the delta features to the feature set improves the results still further (Kovács and Tóth, 2011). Unfortunately, incorporating the delta features into the joint model presented here would be technically challenging, because we would have to propagate the error through the derivatives. However, training the network on several neighbouring spectro-temporal patches instead of just one is possible by modifying the proposed structure so that it has convolutional units (Abdel-Hamid et al., 2012; Vesely et al., 2011). This modification is highlighted in Fig. 3 by boxes being drawn in light grey. As can be seen, in convolutional networks the feature extraction layer performs its operations on several input patches instead of just one. We should also mention here that each input block is processed by the same weights, so the number of weights does not change with the introduction of convolutional processing. Hence, the convolutional techniques allow us to process a much longer time span of input without increasing the number of weights. Note also that the input patches used do not necessarily have to be immediate neighbours, but here we chose this simple scenario.

With the convolutional technology we are able to incorporate the neighbouring patches into the classification process. In an earlier study of this approach, the results of the simple and the convolutional network had been compared, and it had been found that the convolutional network beyond doubt performed better (Kovács and Tóth, 2013). Thus, here we report results only with the convolutional net. But using several input patches during training is not exactly the same as using the Δ and $\Delta\Delta$ coefficients, as we did in the case of the filter selection experiments. The reader should bear this in mind when comparing the results of this section with those of the earlier sections.

7.2. Training from scratch vs. fine-tuning existing spectro-temporal filters

The structure depicted in Fig. 3 allows the algorithm to evaluate the spectro-temporal features and the ANN in one step. However, our main goal here was to extend the scope of the backpropagation algorithm to the feature extraction layer as well. This way, we were also able to train the weights associated with the spectro-temporal filters, and hence fine-tune the initial coefficients. Of course, we had the option to initialize these coefficients randomly (just as we do with all the other weights of the network), but it was also possible to initialize them with some pre-calculated values such as the Gabor coefficients of the *Manual set* found earlier. As the backpropagation algorithm guarantees only a local optimum, initializing the model with weights that already provide a good solution may help the backpropagation algorithm to find a better local optimum than the one found using random initial values.

Here, we experimented with two different initialization schemes for the filter coefficients (i.e. the feature extraction layer of the network). In the first case, they were initialized with random numbers, as they usually are with neural nets. In the second case, the coefficients were initialized based on the coefficients of the *Manual set* (cf. Sect. 4.2).

8. Experiments and results

Below, we describe the experimental settings used for the proposed neural net architecture, and then present and discuss the results got using the different neural net parameter settings.

8.1. Neural net classifier

In the experiments, the classifier we applied was a multilayer neural network specially adapted for this purpose. It consisted of a hidden feature extraction layer with a linear activation function, a hidden layer (with 4,000 neurons) using the sigmoid activation function, and an output layer containing softmax units. The number of output neurons was set to the number of classes (39 in experiments with TIMIT, and 52 in experiments with the Hungarian speech database), while the number of neurons in the feature extraction layer was 54 (9 filters in each of the 6 frequency bands, with one neuron representing each filter). The neural net was trained with random initial weights in the hidden and output layers, using standard backpropagation on 90% of the training data in semi-batch mode, while validation on the remaining, randomly selected

Table IV. Phone recognition accuracy scores got on the clean core test set of TIMIT

Initial Filter weights	Filter weights	
	Unaltered	Trained
Random	72.99%	73.77%
Gabor (manual)	**73.49%**	**74.31%**

10% of the training set was used as the stopping criterion. Like the experiments performed earlier, ten independent neural nets were trained for each parameter configuration, and their average performance was reported.

8.2. Clean speech experiments on TIMIT

Table IV shows the phone recognition results obtained on the clean TIMIT core test set. When evaluating the results, we should focus on two issues. The first is how the expansion of the backpropagation algorithm to the feature extraction layer affects the phone recognition accuracy scores. To learn this, we need to compare the accuracy scores attained using unaltered and trained filter weights. The other important issue is whether initializing the filter weights using the coefficients of the manually found Gabor set helps the backpropagation algorithm find a better solution. To answer this second question, we will compare the results got when the initial filter weights were set randomly with the case where they were initialized based on Gabor filters.

Earlier, the best result was obtained with the manually selected filter set. The most similar configuration in Table IV is the case where the filters are initialized with the Gabor filters and left unaltered during training. Actually, the results are practically the same (73.22% vs. 73.49%), even though the observation context was modelled by both the Δ and $\Delta\Delta$ coefficients and by training the neural net with 9 neighbours in one case and by just the convolutional units in the other. We see again that randomly initializing the filter parameters and not training them at all can give surprisingly good results.

The second column shows the phone accuracy scores got when the backpropagation algorithm refines the weights of the lowest layer as well. The improvements are significant compared to the untrained case, and we also see that initializing the weights in the feature extraction layer according to the manually found Gabor set is also beneficial. The

Table V. Phone recognition accuracy scores got on the noise contaminated test set of TIMIT.)

Noise	Initial Filter weights	20db unaltered	20db trained	10db unaltered	10db trained
Babble	Random	61.70%	62.34%	43.26%	43.77%
	Gabor (manual)	62.58%	**63.05%**	**45.64%**	**45.87%**
Pink	Random	47.75%	55.03%	29.29%	34.58%
	Gabor (manual)	53.40%	**56.87%**	32.16%	**35.97%**
Band limited	Random	54.89%	**60.07%**	44.46%	**46.35%**
	Gabor (manual)	**60.58%**	**60.13%**	45.76%	**47.47%**

best result is obtained by combining the two methods; that is, manual initialization and training, and the best result – 74.31% – which means a 4% relative error reduction compared to the best score we got with filter selection methods. Note that in (Kovács and Tóth, 2013) worse results were reported and smaller differences between results got using random and manual initialization. We think that this difference arises from the fact that here we used much larger networks – 4,000 hidden neurons instead of 1,000 –, and we also tuned the phone insertion penalty parameter of the recognition system, which was not done in this earlier study (Kovács and Tóth, 2013). Working with such a large network increases the risk of overfitting, and a proper initialization becomes more important so as to avoid it.

8.2.1. *Noise contaminated speech experiments on TIMIT*

The neural nets obtained in the previous experiment with their learned parameter values were also evaluated on the noise-contaminated version of the TIMIT core test dataset. To make a comparison of the results between pairs of the four versions of neural nets (applying none, one, or both of the refined initialization and training techniques) in Table V, we arranged them in groups of four, and highlighted the best in each of the four-groups in bold. The scores display a pattern similar to that previously observed in Table V: both the training of filter weights and the initialization of those weights based on the manually found Gabor set improve the phone recognition accuracy scores in noisy environments, but in most cases we get the best results when we combine these techniques. In fact a combination of the two methods brings about a significant improvement in performance both in the babble and pink

Table VI. Phone recognition accuracy scores got on Hungarian speech database

Initial Filter weights	Filter weights	
	Unaltered	Trained
Random	73.06%	74.94%
Gabor (manual)	73.64%	75.25%
Gabor (TIMIT)	**74.90%**	**75.36%**

noise case, while in bandlimited noise case it also brings about some improvement in one case, but the difference here is not significant.

8.2.2. *Clean speech experiments on the Hungarian database*
Next, similar to the automatically found filter sets, we evaluated the cross-database performance of the filters got by using this special neural network. In the first set of experiments we investigated how the filter parameters found in the TIMIT experiments perform on the Hungarian database without training. That is, in these experiments only the upper layers of the network were adapted to the Hungarian database; and to set the feature extraction layer weights we used three scenarios: in the first case they were set randomly, in the second case we used the manually found Gabor set for this purpose, and in the third configuration we used a version of the manually found Gabor filter set that had already been enhanced by the joint optimization method on the TIMIT database. The results of these three experiments are shown in the first column of Table VI. As can be seen, the fine-tuning of the filter set on the TIMIT database did not degrade the performance on the Hungarian database, but even gave a small improvement. This is in marked contrast with the filter selection methods where the filters sets selected based on TIMIT all show performance degradation when migrated to the Hungarian corpus.

While in the case of the filter selection algorithms repeating the whole selection process on a new database would have required days (if not weeks), in this case we could easily run a variant of the previous experiments when the spectro-temporal filters were also adapted to the Hungarian database. The results of these experiments are shown in the second column of Table VI. As can be seen, fine-tuning the filters to the given database can improve the results even further; in fact, for the Hungarian database it is the first setting which provides a better phone recognition accuracy score than those got using the standard MFCCs.

9. Conclusions and Future work

Using Gabor filters to extract spectro-temporal features from speech signals is a relatively new feature extraction method that was motivated by noting the similarity between Gabor filters and the response characteristic of certain cortical neurons. As the neuropsysiological studies do not tell us exactly what the optimal parameters of the Gabor filters are or should be, most authors to date propose the use of automatic feature selection methods such as the FFNN method presented earlier. The filter sets found by these algorithms give a reasonably good performance, comparable to that of standard MFCCs for clean speech and much better for noisy speech. These good results may have given everyone the false impression that the feature selection methods are able to find optimal, or at least near-optimal parameter values. However, in this paper we pointed out that *the good performance originates from the surprising fact that even a randomly selected filter set can yield good results.* This phenomenon is a relatively new observation in image processing, which is currently shifting its research efforts from finding a small but intensively optimised set of basis functions to using larger, overcomplete bases that are chosen near randomly. Earlier, we showed from the results of several experiments why the feature selection strategy used by the algorithms like FFNN and SFFS is not effective in the case of Gabor filters.

As an alternative to the automatic filter selection methods, we used some simple heuristics to manually construct a filter set, which in most cases gave results no worse or even better than those got using the automatically selected sets. However, for this filter set we still have no guarantee of optimality or even local optimality. This can be ensured by some optimization method, and this is why we decided to create a solution that is based on machine learning. Afterwards, we introduced a neural network architecture that can simultaneously perform the optimization of the filter weights and the neural net classifier. We showed by performing several experiments that this method leads to significantly better recognition results than those got via the automatic feature selection algorithms. We also showed that we can still utilize the manually found filter set: it can be used to initialize the backpropagation training process, yielding better results here than when the weights are initialized using conventional random initialization.

In the future, we would like to find a way to incorporate the Δ and $\Delta\Delta$ coefficients into the proposed novel neural net architecture. We also intend to repeat our experiments but this time introduce more hidden layers to our neural network, turning it into a deep neural net. Currently, convolutional deep neural networks provide the best phone

recognition results on TIMIT (Tóth, 2013), and it would be interesting to see whether using special filter selection or filter training methods in their spectro-temporal feature extraction layer can improve their performance still further.

Acknowledgements

This publication is supported by the Eurpean Union and co-funded by the European Social Fund. Project title: Telemedicine-focused research activities in the fields of mathematics, informatics and medical sciences. Project number: TÁMOP-4.2.2.A-11/1/KONV-2012-0073. This study was also supported by the Fund for Scientific Research Flanders (Project "AMODA" G.A122.10N).

References

Aertsen, A. M. and Johannesma, P. I. The spectro-temporal receptive field. A functional characteristic of auditory neurons. *Biological cybernetics, Vol. 42, Issue 2*, pp. 133–143, 1981.

Abdel-Hamid, O., Mohamed, A., Jiang, H., and Penn, G. Applying Convolutional Neural Networks concepts to hybrid NN-HMM model for speech recognition. *Proceedings of ICASSP 2012*, pp. 4277–4280, 2012.

Biem, A., Mcdermott, E. and Katagiri, S. A Discriminative Filter Bank Model For Speech Recognition. *Proceedings of ICASSP-96*, pp. 545–548, 1995.

Bourlard, H. and Morgan, N. Connectionist speech recognition: a hybrid approach. *Kluwer Academic Publication* 1994.

Ezzat, T., Bouvrie, J. and Poggio, T. Spectro-Temporal Analysis of Speech Using 2-D Gabor Filters. *Proceedings of Interspeech* pp. 506–509, 2007.

Gábor, D. Theory of communication *Journal IEE, Vol. 93.*, pp. 429–457, London, 1946.

Gelbart, D., Kleinschmidt, M., and Meyer, B. T. Gabor feature extraction for automatic speech recognition. Retrieved October 22, 2013 from *http://www1.icsi.berkeley.edu/Speech/papers/gabor/*

Gosztolya, G. and Tóth, L. Keyword spotting experiments on broadcast news data using phone-based technologies (in Hungarian). *Proceedings of MSZNY* pp. 224–235, 2010.

Gramss, T. Fast algorithms to find invariant features for a word recognizing neural net. *Proceedings of Second International Conference on Artificial Neural Networks*, pp. 180–184, 1991.

Hirsch, H-G. FaNT: Filtering and Noise-Adding Tool Retrieved March 22, 2010, from *http://dnt.kr.hs-niederrhein.de/download.html*

Huang, G-B., Zhu Q-Y., and Siew, C-K. Extreme learning machine: A new learning scheme of feedforward neural networks. *Proceedings of International Joint Conference on Neural Netwroks*, pp 985–990, 2006.

Huang, L.-L. Shimizu, A. and Kobatake, H. Robust face detection using Gabor filter features. *Pattern Recognition Letters Vol. 26, Issue 11*, pp. 1641–1649, 2005.

Jaitly, N., and Hinton G. Learning a better representation of speech soundwaves using restricted boltzmann machines. *Proceedings of ICASSP 2011*, pp. 5884–5887, 2011.

Jones, JP. and Palmer, LA. An Evaluation of Two-Dimensional Gabor Filter Model of Simple Receptive Fields in Cat Striate Cortex. *Journal of Neurophisiology, Vol. 56, Issue 6*, pp. 1233–1258, 1987.

Kanedera, N., Arai, T., Hermansky, H., and Pavel, M. On the relative importance of various components of the modulation spectrum for automatic speech recognition *Speech Communication, Vol. 28, Issue 1*, pp 43–55, 1999.

Kleinschmidt, M. Methods for capturing spectro-temporal modulations in automatic speech recognition. *Acta Acustica United With Acustica 88, No. 3*, pp. 416–422, 2002a.

Kleinschmidt, M. Spectro-temporal Gabor features as a front end for automatic speech recognition. *Proceedings of Triennial Forum Acusticum 2002* Seville, Spain, September 2002b.

Kleinschmidt, M., Gelbart, D. Improving Word Accuracy with Gabor Feature Extraction. *Proceedings of ICSLP*, pp. 25–28, 2002.

Kovács, G., and Tóth, L. Localized Spectro-Temporal Features for Noise-Robust Speech Recognition. *Proceedings of ICCC-CONTI*, pp. 481–485, 2010

Kovács, G., and Tóth, L. Phone Recognition Experiments with 2D DCT Spectro-Temporal Features. *Proceedings of SACI 2011*, pp. 143–146, 2011.

Kovács, G., and Tóth, L. The Joint Optimization of Spectro-Temporal Features and Neural Net Classifiers. *Proceedings of TSD2013*, pp. 552–559, 2013.

Lamel, L.F., Kassel, R. and Seneff, S. Speech database development: Design and analysis of the acoustic-phonetic corpus. *Proceedings of DARPA Speech Recognition Workshop* pp. 100–109, 1986.

Lee, C., Hyun, D., Choi, E. and Go, J. Optimizing feature extraction for speech recognition *IEEE Transactions on Speech and Audio Processing, Vol. 11*, pp. 80–87, 2003.

Lee, K. F. and Hon, H. W. Speaker-independent phone recognition using Hidden Markov models. *IEEE Trans. Acoust., Speech Signal Processing, Vol. 37*, pp. 1641–1648, Nov. 1989.

Lee, S-M., Fang, S-H., Hung,J-W and Lee L-S. Improved MFCC feature extraction by PCA-optimized filter-bank for speech recognition. *IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU '01*, pp. 49–52, 2001.

Meyer, B. T. and Kollmeier B. Optimization and evaluation of Gabor feature sets for ASR. *Proceedings of Interspeech* pp. 906–909, 2008.

Mohamed, A., Dahl, G.E., and Hinton, G. Acoustic Modeling Using Deep Belief Networks. *IEEE Transactions on Audio, Speech, and Language Processing, Volume 20, Issue 1*, pp. 14–22, 2012.

Palaz, D., Collobert, R., Magimai-Doss, M. End-to-end Phoneme Sequence Recognition using Convolutional Neural Networks. *NIPS Deep Learning Workshop*, 2013.

Sainath, T. N., Kingsbury, A., Ramebhadran, B., Ramebhadran M. Learning Filter Banks within a Deep Neural Network. *Proceedings of ASRU 2013*, 2013.

Schädler M. R., Meyer B. T, and Kollmeier B. Spectro-temporal modulation subspace-spanning filter bank features for robuest automatic speech recognition. *The Journal of Acoustical Society of America*, pp. 4134–4151, 2012.

Somol, P., Novovicova, J. and Pudil, P. Efficient Feature Subset Selection and Subset Size Optimization. In E. Herout editor, *Pattern Recognition Recent Advances* pp. 76–98, InTech, 2010.

Sun, Z., Bebis, G. and Miller, R. Evolutionary Gabor Filter Optimization with Application to Vehicle Detection. *Proceedings of ICDM*, pp. 307–314, 2003.

Tasi, D. M. Optimal Gabor filter design for texture segmentation using stochastic optimization. *Image and Vision Computing, Vol. 19*, pp. 299–316, 2009.

Tiitinen, H., Miettinen, I., Alku, P. and May P. J. C. Transient and sustained cortical activity elicited by connected speech of varying intelligibility. *Neuroscience* 2012.

Tóth, L. Convolutional Deep Rectifier Neural Nets for Phone Recognition. *Proceedings of Interspeech, 2013*, pp. 1722-1726, 2013.

Varga, A. and Steeneken, H. Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication, Vol. 12, No. 3* pp. 247–251, 1993.

Vesely, K., Karafiat, M., and Grezl, F. Convolutive Bottleneck Network features for LVCSR. *Proceedings of ASRU 2011*, pp. 42–47, 2011.

Vinyals, O., and Deng, L. Are Sparse Representations Rich Enough for Acoustic Modeling? *Proceedings of Interspeech 2012*, pp. 1–1, 2012.

von Ossietzky, C. Gabor filter bank features Retrieved September 15, 2013 from `http://medi.uni-oldenburg.de/GBFB`

Young, S. J. and Evermann, G and Gales, M. J. F. and Kershaw, D. and Moore, G. and Odell, J. J. and Ollason, D. G. and Povey, D. and Valtchev, V. and Woodland, P. The HTK book version 3.4. Cambridge University Engineering Department, Cambridge, UK, 2006.